

Edge Impulse Wake Word Model Training | Tech Roundup

Today, we will walk you step by step through learning **how to train a voice keyword model and deploy it to embedded hardware, using the Edgi-Talk platform to adapt Edge Impulse**. Of course, the principles also apply generally to other ARM embedded platforms. Let's see how to get Edgi-Talk started with edge machine learning!

Edge Impulse Introduction

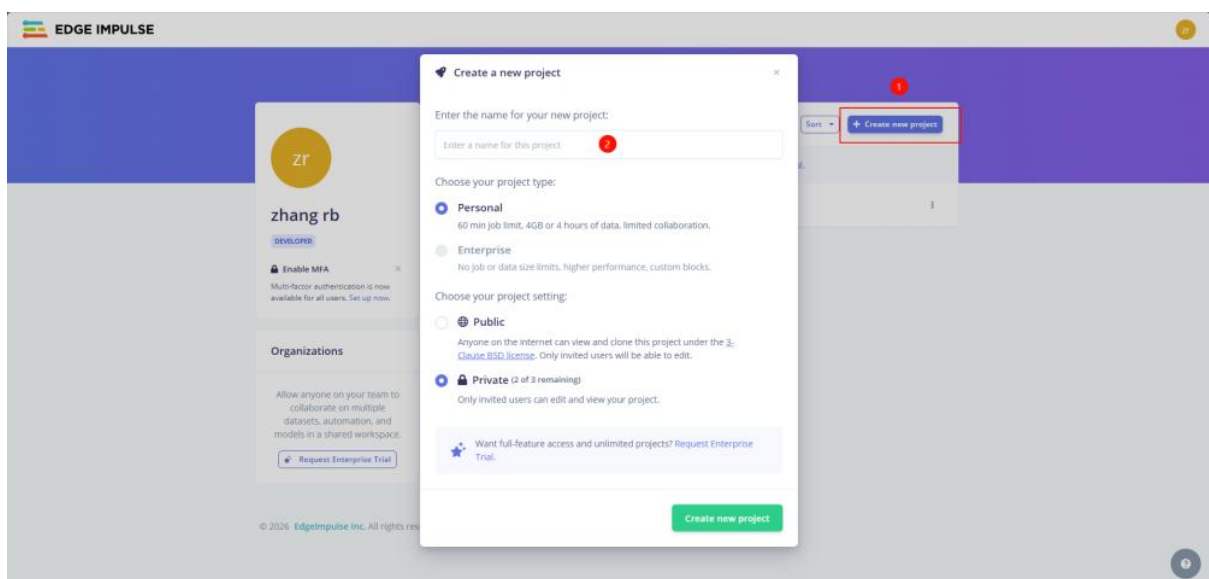
Edge Impulse is the leading-edge AI development platform, dedicated to helping developers, engineers, and enterprises quickly deploy smart AI to any edge device (such as microcontrollers, sensors, and IoT hardware).

You can easily **collect sensor data, build datasets, train and optimize machine learning models, then deploy with one click to thousands of hardware ecosystems worldwide**, without needing deep AI expertise to achieve rapid prototyping and commercialization.

1 Create an Account

Steps:

1. Log in to Edge Impulse, <https://edgeimpulse.com/>
2. Register an account;
3. Via create new project create a project



2 Record Dataset

1. Flash Edgi-Talk dev board with the UAC firmware, used for collecting audio data: uac-firmware.hex; (Link: https://github.com/RT-Thread-Studio/sdk-bsp-psoc_e84-edgi-talk/releases/download/1.1.0/uac-firmware.hex)

Steps to flash firmware via ModusToolbox Programmer

1. Launch the app

"C:\Infineon\Tools\ModusToolboxProgtools-1.9\mtb-programmer\mtb-programmer.exe"

2. **Connect the board** — plug the USB cable into the **DAP** port (this is what powers the board and lets the programmer talk to it).

3. **Check the top settings bar** — it should auto-populate:

- Programmer/Board ID: KIT_PSE84_EVAL_EPC2-BULK-...
- Board Name: KIT_PSE84_EVAL_EPC2
- Device: PSE846GPS2DBZC4A

4. **Click "Connect"** (top toolbar) to establish the connection to the board.

5. **Open the Settings panel** (should show automatically, or click the gear icon) and set:

- **File** → browse to your .hex firmware file (e.g. C:\Users\User\Downloads\uac-firmware.hex)
- **Reset Chip** → ☒ ticked
- **External Memory** → ☒ **ticked (this was the critical step)** — without this, the tool won't set up the external QSPI flash region and the write will silently fail with "wrote 0 bytes"
- **Flashloader** → leave as "Default" (it auto-selects the right one once External Memory is ticked)

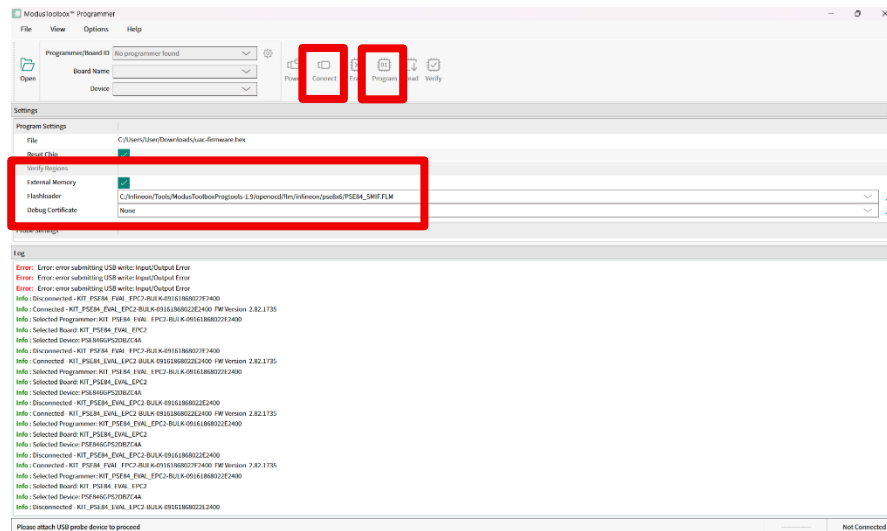
6. **Click "Program"** in the top toolbar.

7. **Check the Log panel at the bottom** for success — look for:

- Info : flash 'cmsis_flash' found at 0x60000000 (confirms the QSPI bank was set up)
- Info : wrote XXXXX bytes from file ... where XXXXX is a real nonzero number

- Info : ** Programming Finished **

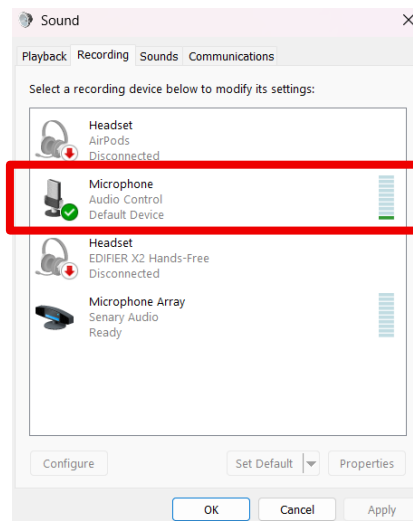
8. **Unplug the DAP cable, plug the cable into the "USB" port instead** — this is the board's application-side data port, separate from the programmer port, and it's what shows up as a working audio device to your PC once the new firmware is running.



2. Verifying the board shows up as a microphone

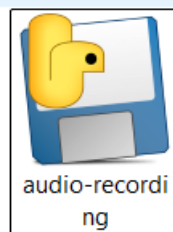
1. **Make sure the cable is plugged into the "USB" port** on the board (not DAP) — this only works with the application-side port connected.
2. Open **Sound settings** via Control Panel:
 - Right-click the speaker icon in your Windows taskbar → **Sound settings** (or **Sounds**), OR
 - Open **Control Panel** → **Sound**
3. Click the **"Recording"** tab.

4. Look for a new entry: For example, Microphone



Confirm it is picking up audio: with that device selected, speak or make noise near the board — you should see the small level meter (the vertical bars next to it) move/light up in response, confirming live audio input.

3. Download and open the audio dataset recording software: Audio-recording.exe; (**Link:** https://github.com/RT-Thread-Studio/sdk-bsp-psoc_e84-edgi-talk/releases/download/1.1.0/audio-recording.exe)



4. For the audio input device, select: Audio Control (MME) device;

唤醒词录制工具 - 数据采集

录音参数设置

采样率: 16000 Hz

录制时长: 10.0 秒

音频设备

输入设备: 麦克风 (Audio Control) (MME) 刷新

文件保存设置

保存目录: C:\Users\RTT\AppData\Local\Temp\MEI234042\data\xiaorui 浏览...

文件名前缀: xiaorui

当前序号: 1 (下一个保存的文件序号)

文件名预览: xiaorui_001.wav

开始录制 保存录音

准备就绪, 请点击开始录制

当前配置: 采样率 16000 Hz, 录制时长 10.0 秒

5. Click Start Recording to make one recording lasting up to 10 seconds; ensure clear speech during recording — wake word recordings need pauses of 1-2s between repetitions;

6. After completing one recording, click on the right side "Save Recording" option to save it;

7. The current sequence number will increment by default; repeat the above steps to record 20-50 keyword audio files;

8. Ultimately, record two types of audio files: one for the wake word and one for noise;

桌面 > audio训练 > data >

  排序  查看 

- ☐ 名称
- noise
 - xiaorui

桌面 > audio训练 > data > noise

  排序  查看 

☐ 名称 # 标题

-  noice_001.wav
-  noice_002.wav
- ☐  noice_003.wav
-  noice_004.wav
-  noice_005.wav
-  noice_006.wav
-  noice_007.wav
-  noice_008.wav
-  noice_009.wav
-  noice_010.wav
-  noice_011.wav
-  noice_012.wav

桌面 > audio训练 > data > xiaorui

  排序  查看 

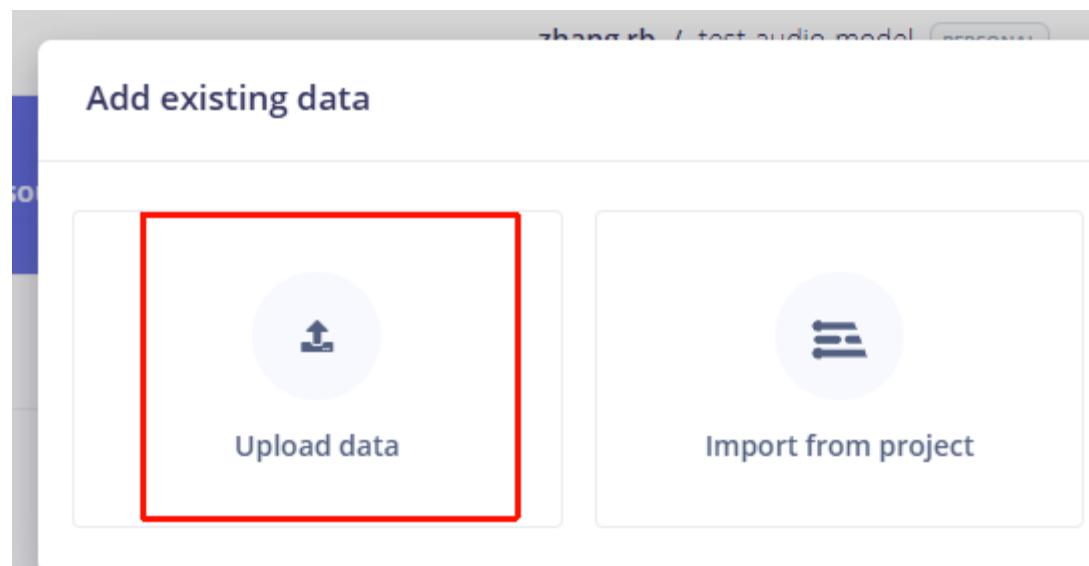
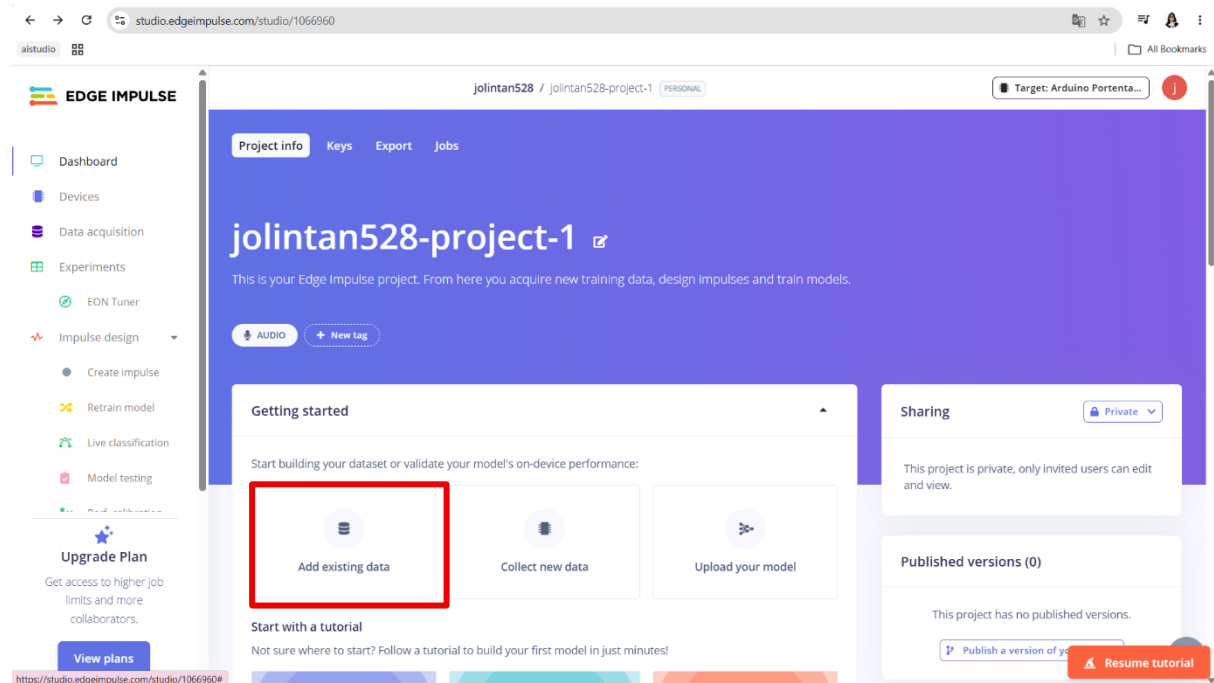
☐ 名称 # 标题

-  xiaorui_001.wav
-  xiaorui_002.wav
-  xiaorui_003.wav
-  xiaorui_004.wav
-  xiaorui_005.wav
-  xiaorui_006.wav
-  xiaorui_007.wav
-  xiaorui_008.wav
-  xiaorui_009.wav
-  xiaorui_010.wav
-  xiaorui_011.wav
-  xiaorui_012.wav

3 Data Upload

1. Click **add existing data**

2. Select upload data



3. Select and upload the audio folder on your computer; the site will automatically classify the files based on their filenames (noise, wake word)

Upload data

You can upload CBOR, JSON, CSV, Parquet, WAV, JPG, PNG, AVI or MP4 files. You can also upload an annotation file named "info.labels" with your data to assign bounding boxes, labels, and/or metadata. View [Uploader docs](#) to learn more. Alternatively, you can use our [Python SDK](#) to programmatically ingest data in various formats, such as pandas or numpy; or [import data directly from another project](#).

For CSV and Parquet files, [configure the CSV Wizard](#) to define how your files should be processed before uploading files.

Upload mode

- ☐ Select individual files ?
- ☒ Select a folder ? 1

Select files

选择文件 未选择文件

Choose Folder

Upload into category

- ☒ Automatically split between training and testing ? 2
- ☐ Training
- ☐ Testing

Label

- ☒ Infer from filename ? 3
- ☐ Leave data unlabeled ?
- ☐ Enter label:

Enter a label

Upload output

```
[noise] Uploading 17 files...
[19/35] Uploading noise_002.wav OK
[20/35] Uploading noise_003.wav OK
[21/35] Uploading noise_009.wav OK
[22/35] Uploading noise_011.wav OK
[23/35] Uploading noise_005.wav OK
[24/35] Uploading noise_006.wav OK
[25/35] Uploading noise_017.wav OK
[26/35] Uploading noise_010.wav OK
[27/35] Uploading noise_012.wav OK
[28/35] Uploading noise_008.wav OK
[29/35] Uploading noise_013.wav OK
[30/35] Uploading noise_001.wav OK
[31/35] Uploading noise_004.wav OK
[32/35] Uploading noise_007.wav OK
[33/35] Uploading noise_014.wav OK
[34/35] Uploading noise_016.wav OK
[35/35] Uploading noise_015.wav OK
```

Done. Files uploaded successful: 35. Files that failed to upload: 0.

Job completed

< Back

Upload data

You can upload CBOR, JSON, CSV, Parquet, WAV, JPG, PNG, AVI or MP4 files. You can also upload an annotation file named "info.labels" with your data to assign bounding boxes, labels, and/or metadata. View [Uploader docs](#) to learn more. Alternatively, you can use our [Python SDK](#) to programmatically ingest data in various formats, such as pandas or numpy; or [import data directly from another project](#).

For CSV and Parquet files, [configure the CSV Wizard](#) to define how your files should be processed before uploading files.

Upload mode

- ☐ Select individual files ?
- ☒ Select a folder ?

Select files

Choose Files No file chosen

Choose Folder

Upload into category

- ☒ Automatically split between training and test ?
- ☐ Training
- ☐ Test

Label

- ☒ Infer from filename ?
- ☐ Leave data unlabeled ?
- ☐ Enter label:

Enter a label

Upload output

```
[ 4/21] Uploading xiaorui_019.wav OK
[ 5/21] Uploading xiaorui_002.wav OK
[ 6/21] Uploading xiaorui_011.wav OK
[ 7/21] Uploading xiaorui_008.wav OK
[ 8/21] Uploading xiaorui_016.wav OK
[ 9/21] Uploading xiaorui_007.wav OK
[10/21] Uploading xiaorui_010.wav OK
[11/21] Uploading xiaorui_015.wav OK
[12/21] Uploading xiaorui_012.wav OK
[13/21] Uploading xiaorui_020.wav OK
[14/21] Uploading xiaorui_006.wav OK
[15/21] Uploading xiaorui_009.wav OK
[16/21] Uploading xiaorui_005.wav OK
[17/21] Uploading xiaorui_014.wav OK
[18/21] Uploading xiaorui_017.wav OK
[19/21] Uploading xiaorui_018.wav OK
[20/21] Uploading xiaorui_013.wav OK
[21/21] Uploading xiaorui_021.wav OK
```

Done. Files uploaded successful: 21. Files that failed to upload: 0.

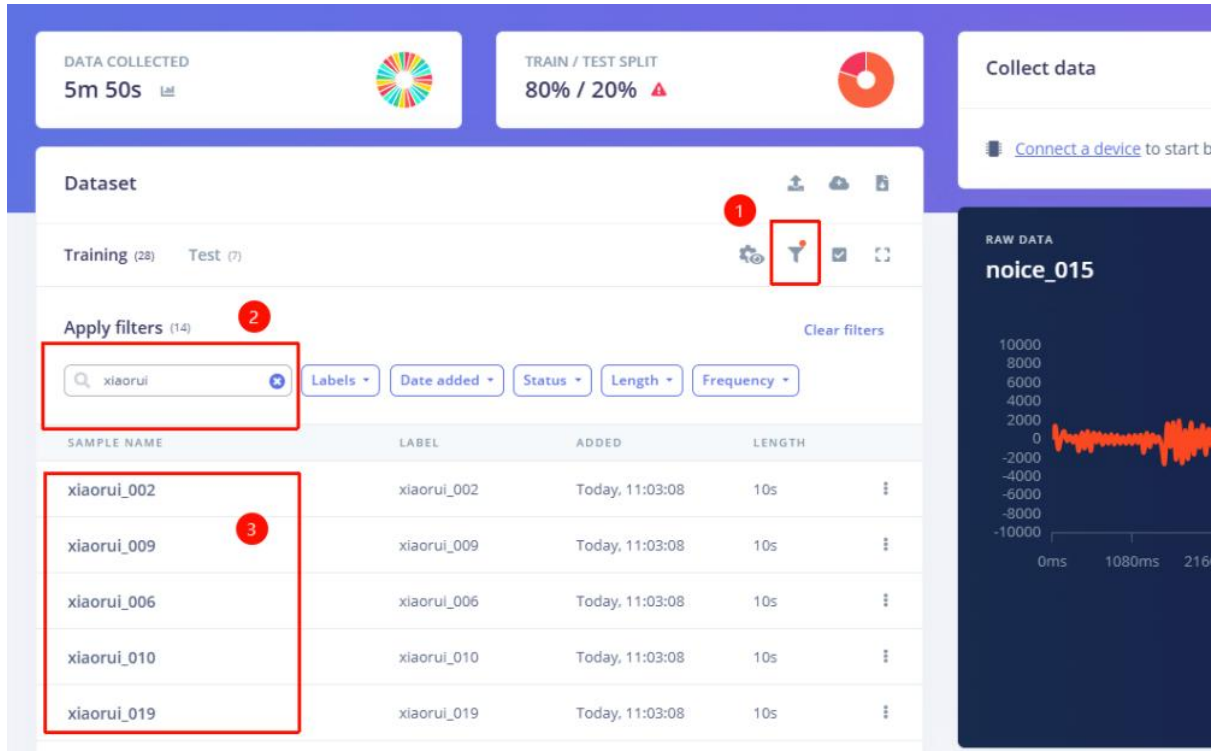
Job completed

xiaorui folder

4 Data Splitting

Steps:

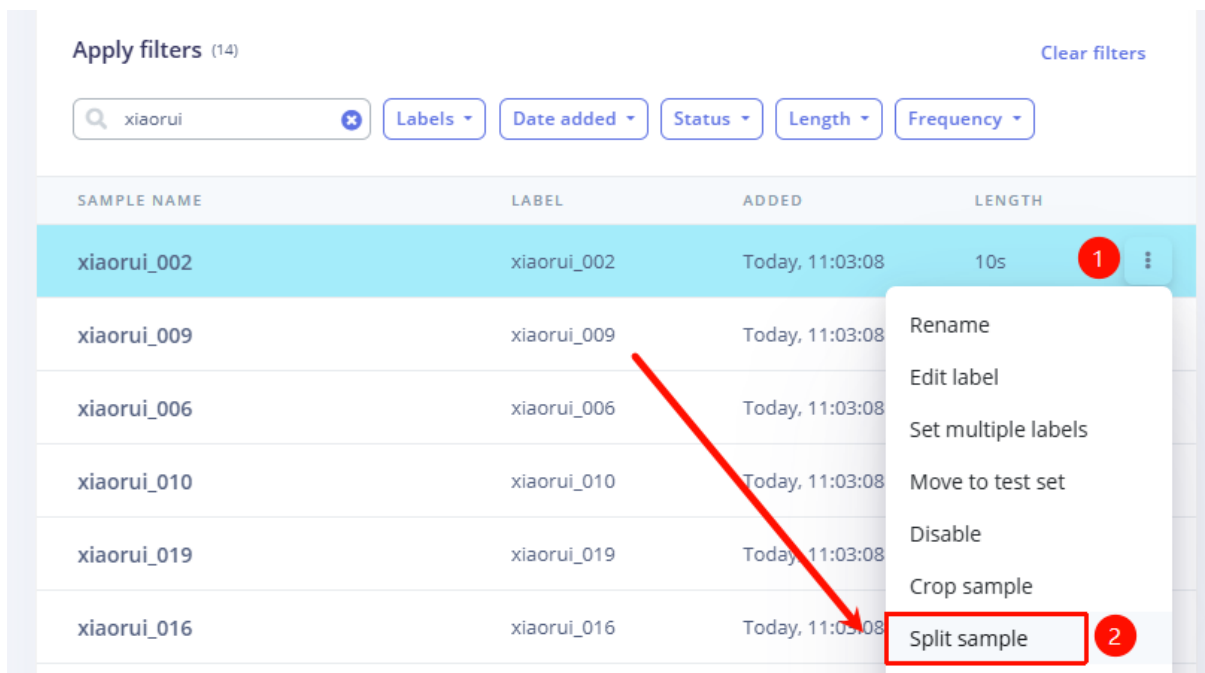
1. First select the Training list data, click Filter first, and enter our **wake word** file prefix to filter out the files we want to work with.



The screenshot shows the 'Dataset' section of a platform. At the top, there are statistics: 'DATA COLLECTED 5m 50s' and 'TRAIN / TEST SPLIT 80% / 20%'. Below these, the 'Dataset' tab is active, showing 'Training (28)' and 'Test (7)' counts. A red box labeled '1' highlights the 'Filter' icon. Below the filter icon, a red box labeled '2' highlights the 'Apply filters (14)' section, which contains a search bar with 'xiaorui' and several filter buttons. A red box labeled '3' highlights the first row of the table, 'xiaorui_002'. The table has columns: 'SAMPLE NAME', 'LABEL', 'ADDED', and 'LENGTH'. To the right, there is a 'RAW DATA' section showing a waveform for 'noise_015'.

SAMPLE NAME	LABEL	ADDED	LENGTH
xiaorui_002	xiaorui_002	Today, 11:03:08	10s
xiaorui_009	xiaorui_009	Today, 11:03:08	10s
xiaorui_006	xiaorui_006	Today, 11:03:08	10s
xiaorui_010	xiaorui_010	Today, 11:03:08	10s
xiaorui_019	xiaorui_019	Today, 11:03:08	10s

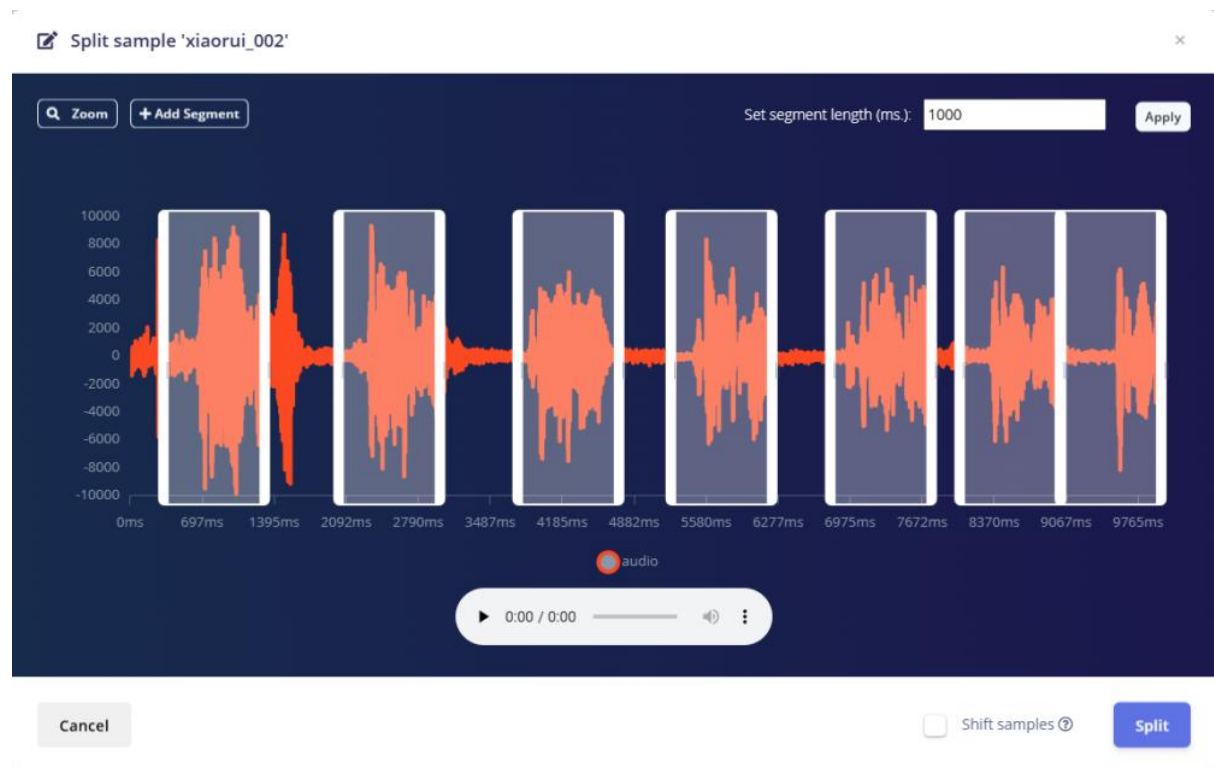
2. After clicking the audio file on the right, select **Split sample**, and the platform will split the audio file into roughly 1s audio segments



The screenshot shows the 'Apply filters (14)' section of the dataset interface. A red box labeled '1' highlights the first row of the table, 'xiaorui_002'. A red arrow points from this row to a context menu that is open. The context menu has several options: 'Rename', 'Edit label', 'Set multiple labels', 'Move to test set', 'Disable', 'Crop sample', and 'Split sample'. A red box labeled '2' highlights the 'Split sample' option.

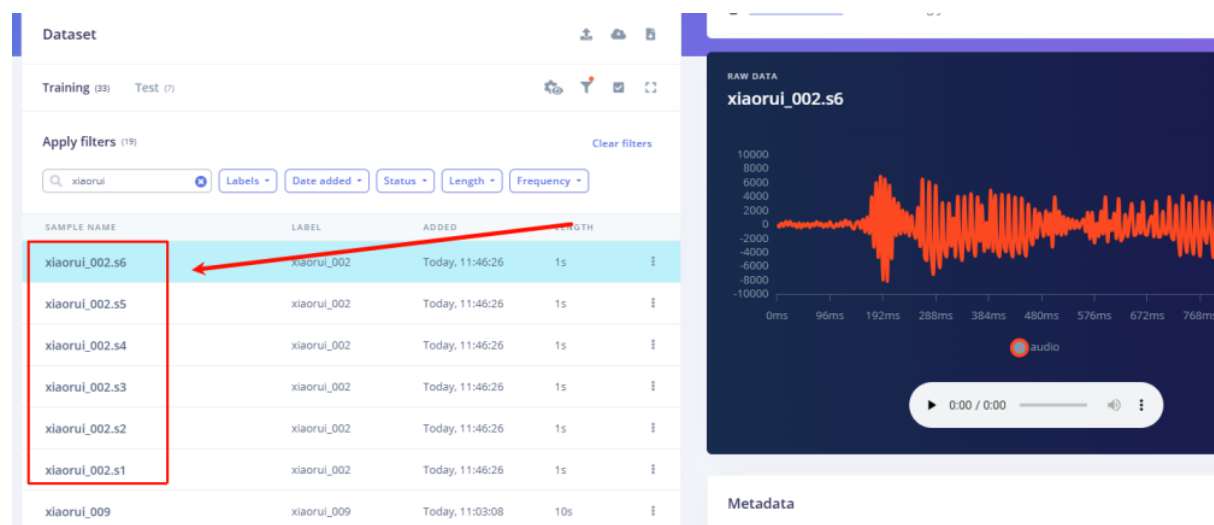
SAMPLE NAME	LABEL	ADDED	LENGTH
xiaorui_002	xiaorui_002	Today, 11:03:08	10s
xiaorui_009	xiaorui_009	Today, 11:03:08	10s
xiaorui_006	xiaorui_006	Today, 11:03:08	10s
xiaorui_010	xiaorui_010	Today, 11:03:08	10s
xiaorui_019	xiaorui_019	Today, 11:03:08	10s
xiaorui_016	xiaorui_016	Today, 11:03:08	10s

3. On the audio splitting screen, you can adjust the audio segments to cover the wake word audio content and play them back to listen and adjust.



Each rectangle represents a sub-segment extracted by the split. If necessary, you can adjust the position of the rectangle so it fully covers our wake word audio area, or add/remove segments as needed.

4. After splitting, the audio files become independent files, and the site will automatically add a suffix (xxxx_Sn)



5. Following the steps above, split all the wake word files

6. Select the Test list data; for the wake word files continue splitting; noise labelled files don't need to be split.

The screenshot shows a dataset management interface with a blue header. The header contains tabs: Dataset, Data explorer, Data sources, Synthetic data, AI labeling (NEW), and CSV Wizard. Below the header, there are two summary cards. The first card, 'DATA COLLECTED', shows '4m 31s' and a colorful circular progress indicator. The second card, 'TRAIN / TEST SPLIT', shows '74% / 26%' and a red circular progress indicator. Below these cards is a 'Dataset' section. It has two tabs: 'Training (74)' and 'Test (7)'. The 'Test (7)' tab is selected and highlighted with a red box. A red circle with the number 1 is next to the 'Test (7)' tab. Below the tabs is a table with columns: SAMPLE NAME, LABEL, ADDED, and LENGTH. The table contains seven rows of data. A red box highlights the first four rows of the table, which are: xiaorui_012, xiaorui_007, xiaorui_011, and xiaorui_001. A red circle with the number 2 is next to the second row (xiaorui_007). At the bottom right of the table, there are navigation buttons: a left arrow, a button with the number 1, and a right arrow.

SAMPLE NAME	LABEL	ADDED	LENGTH
noice_001	noice_001	Today, 11:03:10	10s
noice_017	noice_017	Today, 11:03:10	10s
noice_003	noice_003	Today, 11:03:10	10s
xiaorui_012	xiaorui_012	Today, 11:03:08	10s
xiaorui_007	xiaorui_007	Today, 11:03:08	10s
xiaorui_011	xiaorui_011	Today, 11:03:08	10s
xiaorui_001	xiaorui_001	Today, 11:03:08	10s

7. Classify the data: following the steps in the image below, divide the dataset into two main categories: Wake Word + Other.

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Experiments

EON Tuner

Impulse design

Create impulse

MFCC

Classifier

Retrain model

Live classification

Model testing

Perf. calibration

Upgrade Plan

Get access to higher job limits and more collaborators.

DATA COLLECTED

10m 38s

TRAIN / TEST SPLIT

80% / 20%

Collect data

Connect a device

RAW DATA

Click on a sam

Dataset

Training (197) Test (50)

Apply filters (162) Clear filters

1

Labels

Date added

Status

Length

Frequency

2

3

Delete 162

Edit labels 162

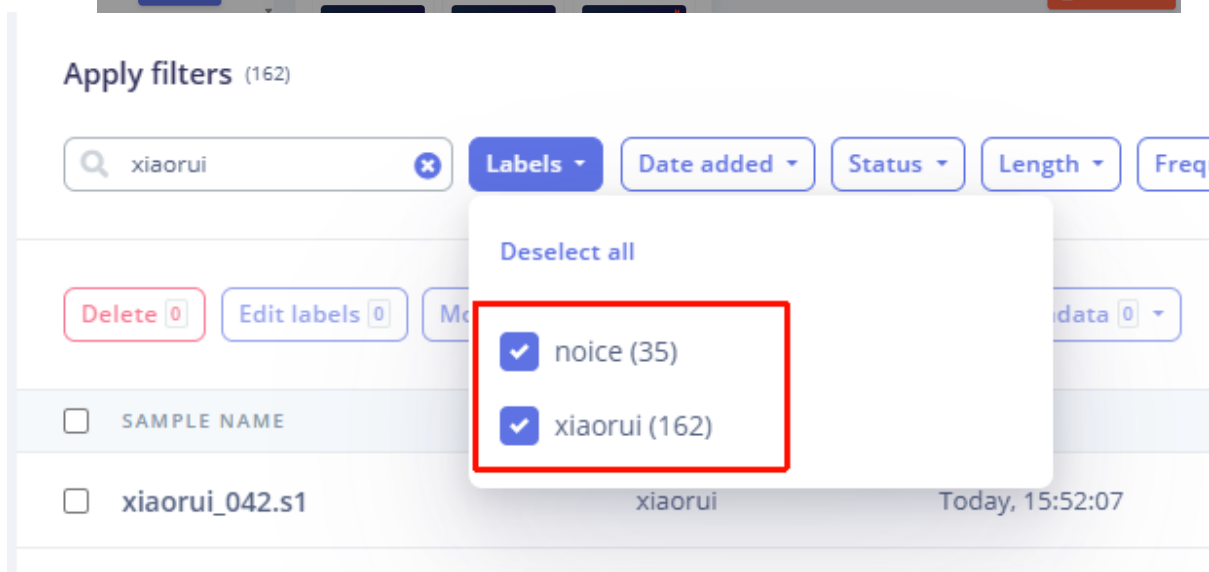
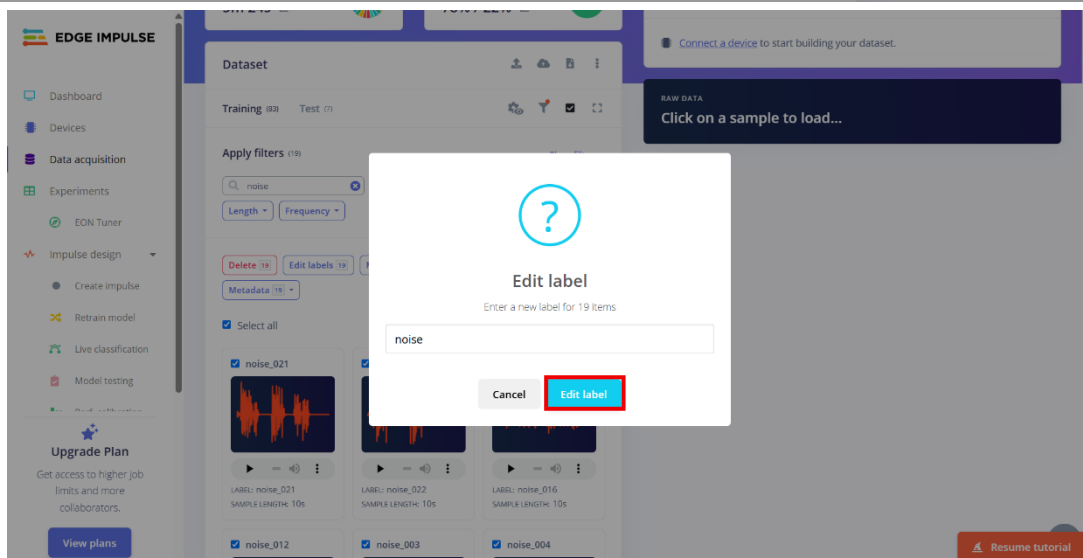
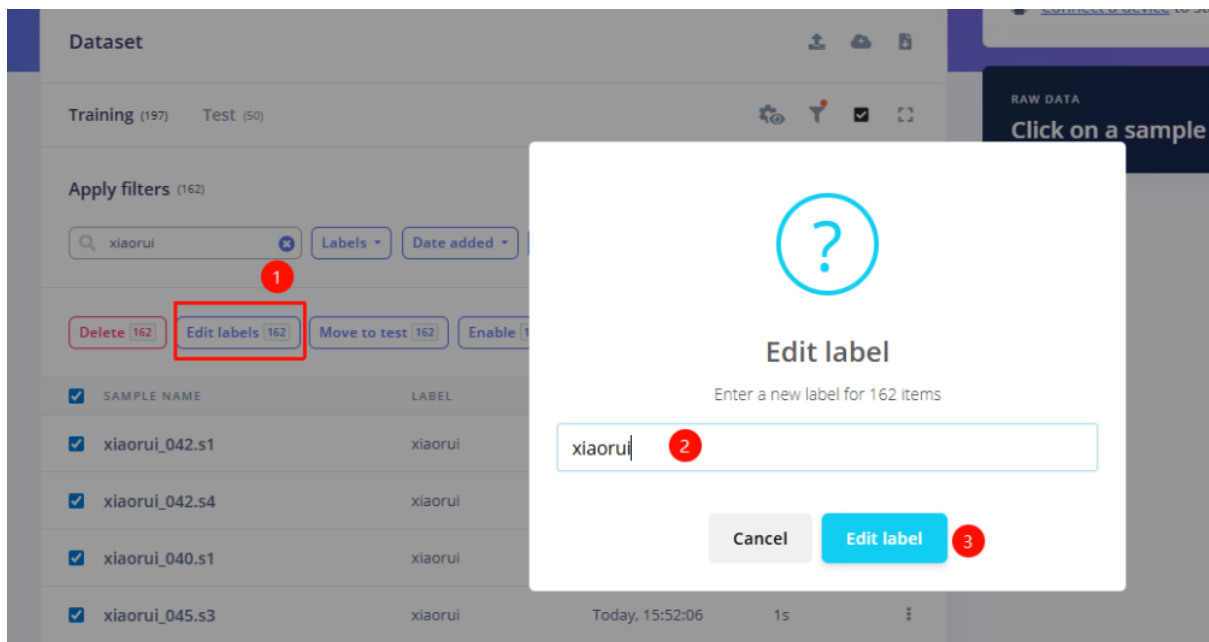
Move to test 162

Enable 162

Disable 162

Metadata 162

☒	SAMPLE NAME	LABEL	ADDED	LENGTH
☒	xiaorui_042.s1	xiaorui	Today, 15:52:07	1s
☒	xiaorui_042.s4	xiaorui	Today, 15:52:07	1s
☒	xiaorui_040.s1	xiaorui	Today, 15:52:06	1s
☒	xiaorui_045.s3	xiaorui	Today, 15:52:06	1s
☒	xiaorui_047.s2	xiaorui	Today, 15:52:06	1s
☒	xiaorui_040.s5	xiaorui	Today, 15:52:06	1s
☒	xiaorui_042.s2	xiaorui	Today, 15:52:06	1s
☒	xiaorui_042.s3	xiaorui	Today, 15:52:06	1s



5 Model Training

5.1 Create Impulse (Preprocessing/Model Definition)

An "Impulse" here is the term used by **Edge Impulse** to describe the data processing—training pipeline.

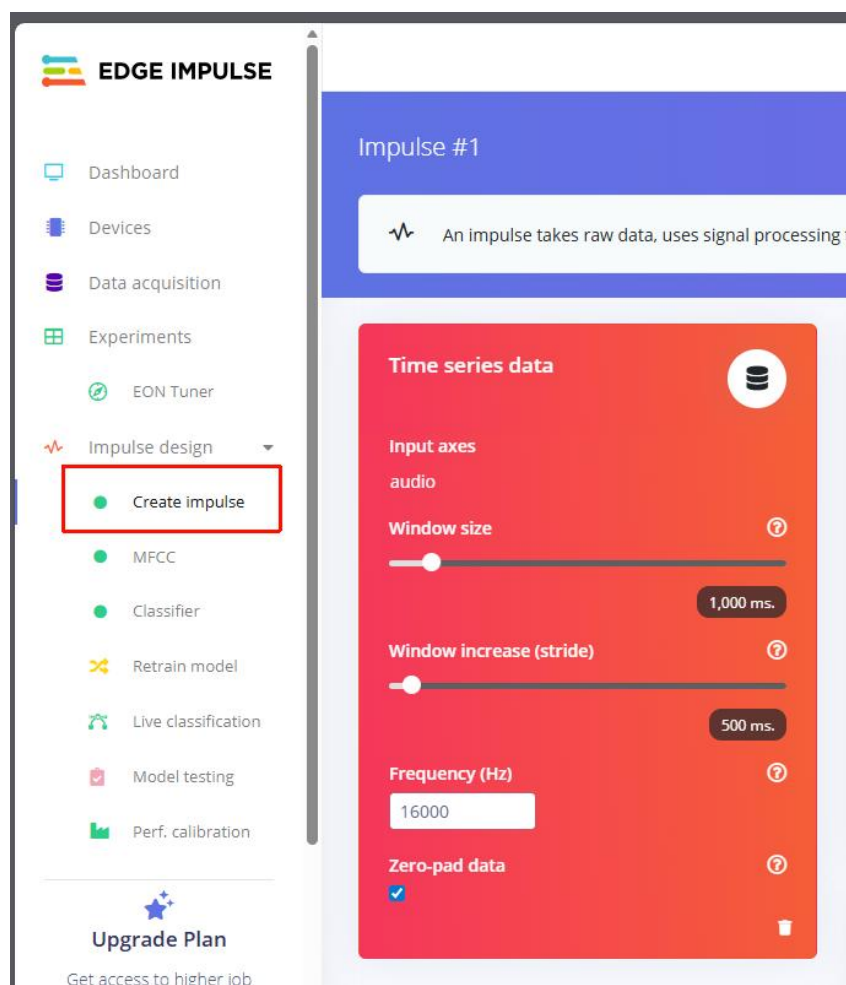
Create the impulse and set the window size to 1000 ms, and set the window increase to 500 ms (overlapping windows to augment the data), with the frequency set to 16KHz.

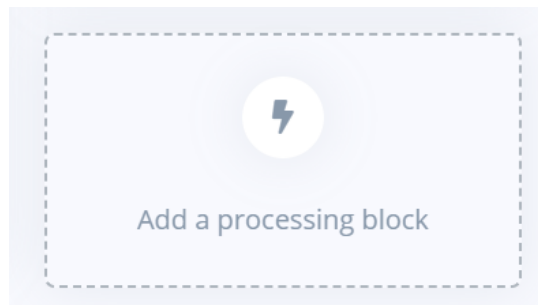
These settings mean that each time inference is run, sensor data collection will complete within 1000 ms, with the exact number of samples determined by the sampling frequency.

In short, your device will, within a 1000 ms time window, collect N data samples, then preprocess these samples and feed them into the neural network to obtain the inference result.

Steps:

1. Click on the left **Create impulse**
2. Then click **Add a processing block to** add Audio(MFCC)





⚡ Add a processing block

×

Did you know? You can [bring your own DSP code](#).

DESCRIPTION	AUTHOR	RECOMMENDED
Audio (MFCC) OFFICIALLY SUPPORTED Extracts features from audio signals using Mel Frequency Cepstral Coefficients, great for human voice.	Edge Impulse	★ <button>Add</button>
Audio (MFE) OFFICIALLY SUPPORTED Extracts a spectrogram from audio signals using Mel-filterbank energy features, great for both voice and non-voice audio.	Edge Impulse	★ <button>Add</button>
Spectrogram OFFICIALLY SUPPORTED Extracts a spectrogram from audio or sensor data, great for non-voice audio or data with continuous frequencies.	Edge Impulse	<button>Add</button>
Audio (Syntiant) OFFICIALLY SUPPORTED Syntiant only. Compute log Mel-filterbank energy features from an audio signal.	Syntiant	<button>Add</button>
Raw Data OFFICIALLY SUPPORTED Use data without pre-processing. Useful if you want to use deep learning to learn features.	Edge Impulse	<button>Add</button>

Some processing blocks have been hidden based on the data in your project. [Show all blocks anyway](#)

3. A dialog box will pop up below; the default configuration is fine:

Configure your target device and application budget

Target device
Define your target device requirements to inform model optimizations and performance calculations. No device yet? Use the default settings which you can change at any time.

Target device: Cortex-M7 216MHz

Processor family: Cortex-M

Clock rate: 216 MHz

Custom device name (optional):

Application budget
Specify the available RAM and ROM for the model's operation, along with the maximum allowed latency for your specific application. Not sure yet? Start with the defaults and modify them later on.

RAM: 340 KB

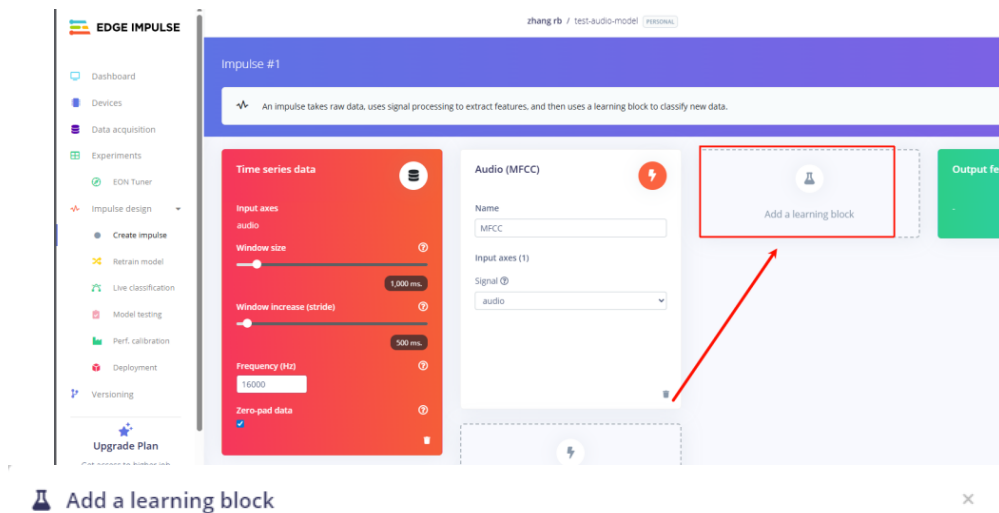
ROM: 1 MB

Latency: 100 ms

Reset to default settings Cancel Save

4. Use **MFCC**, which uses Mel-Frequency Cepstral Coefficients to extract features from the audio signal — very useful for human speech.

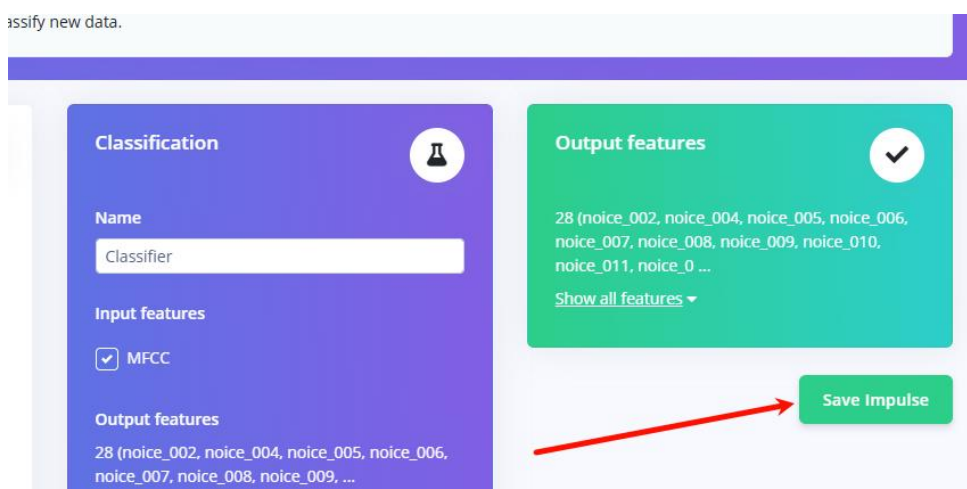
5. Then click **Add a learning block** to add the **Classification** block, which builds our model from scratch using a convolutional neural network for image classification.



Did you know? You can [bring your own model](#) in PyTorch, Keras or scikit-learn.

DESCRIPTION	AUTHOR	RECOMMENDED
Classification OFFICIALLY SUPPORTED Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.	Edge Impulse	★ Add
Regression OFFICIALLY SUPPORTED Learns patterns from data, and can apply these to new data. Great for predicting numeric continuous values.	Edge Impulse	Add
Transfer Learning (Keyword Spotting) OFFICIALLY SUPPORTED Fine tune a pre-trained keyword spotting model on your data. Good performance even with relatively small keyword datasets.	Edge Impulse	Add

6. Finally, click save impulse to save the configuration



5.2 Preprocessing (MFCC)

MFCC is a widely used method for converting audio signals into 2D features representing speech frequency patterns, which are well suited for speech-based recognition models.

We create spectrogram images from the recorded audio.

Steps:

1. Click **MFCC**, we can keep the default parameter values.
2. Click **Save parameters**.

The screenshot displays the EDGE IMPULSE software interface. On the left sidebar, the 'MFCC' option is highlighted with a red box and a red circle containing the number '1'. The main panel is titled 'Parameters' and contains a section for 'Mel Frequency Cepstral Coefficients'. This section includes input fields for 'Number of coefficients' (13), 'Frame length' (0.02), 'Frame stride' (0.02), 'Filter number' (32), 'FFT length' (256), 'Normalization window size' (101), 'Low frequency' (0), and 'High frequency' (Click to set). Below this is a 'Pre-emphasis' section with a 'Coefficient' field set to 0.98. A red box with a red circle containing the number '2' highlights the 'Save parameters' button at the bottom right of the parameter section. To the right of the parameter section, there is a 'Raw features' section showing a list of numerical values and a 'Label' field with the value 'xiaorui_007'. Further right, the 'DSP result' section shows a 'Cepstral Coefficients' spectrogram and 'Processed features' with a list of numerical values. At the bottom right, the 'On-device performance' section shows a 'PROCESSING TIME' of 85 ms.

3. Click **Generate Features** to generate 3 labels' worth of feature data.

Parameters

Generate features

Training set

Data in training set	3m 7s
Classes	28 (noice_002, noice_004, noice_005, noice_006, noice_007, noice_008, noice_009, noice_010, noice_011, noice_012, noice_013, noice_014, noice_015, noice_016, xiaorui_002, xiaorui_003, xiaorui_004, xiaorui_006, xiaorui_007, xiaorui_008, xiaorui_009, xiaorui_010, xiaorui_013, xiaorui_014, xiaorui_015, xiaorui_016, xiaorui_018, xiaorui_019)
Training windows	313

Generate features

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Experiments

EON Tuner

Impulse design

Create impulse

MFCC

Classifier

Retrain model

Upgrade Plan

View plans

Parameters

Generate features

Training set

Data in training set	4m 14s
Classes	2 (noise, xiaorui)
Training windows	425

Generate features

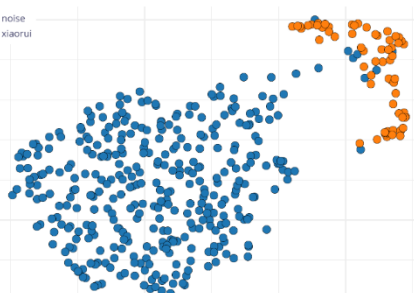
Feature generation output

[16/16] Pre-caching files...
Pre-caching files OK
[1/83] Creating windows from files...
[69/83] Creating windows from files...
[83/83] Creating windows from files...
Created 425 windows: noise: 361, xiaorui: 64
Creating features...
[1/425] Creating features...
[425/425] Creating features...
Created features
Job completed (success)

Feature explorer

noise

xiaorui



On-device performance

PROCESSING TIME

19 ms.

PEAK RAM USAGE

15 KB

Resume tutorial

https://studio.edgeimpulse.com/studio/1066960/impulse/1/dsp/mfcc/5

5.3 Model Design and Training (Classifier)

Next, we need to design the model structure and begin training. Steps are as follows:

Steps:

1. Click on the left **Classifier**, the entire model structure design has already been configured.
2. Then click **Save & Train** to start training the model.

The screenshot displays the EON Tuner interface for the Classifier model. The left sidebar shows the navigation menu with 'Classifier' highlighted. The main panel is divided into 'Neural Network settings' and 'Training output'.

Neural Network settings:

- Training settings:**
 - Number of training cycles: 300
 - Use learned optimizer: ☐
 - Learning rate: 0.005
 - Training processor: GPU
- Advanced training settings:**
 - Audio training options:**
 - Data augmentation: ☒
 - Add noise: None Low High
 - Mask time bands: None Low High
 - Mask frequency bands: None Low High
 - Warp time axis: ☐
- Neural network architecture:**
 - Neural network
 - Transfer learning
 - Load preset...

Training output:

9: UndefinedMetricWarning: Only one class is present in y_true. ROC AUC score is not defined in that case.
Warnings.warn(
Calculating int8 accuracy...
/app/keras/.venv/lib/python3.10/site-packages/sklearn/metrics/_ranking.py:37
9: UndefinedMetricWarning: Only one class is present in y_true. ROC AUC score is not defined in that case.
Warnings.warn(
Extracting TensorBoard logs...
Extracting TensorBoard logs OK
Model training complete
Job completed (success)

Model: Model version: Quantized (int8)

Last training performance (validation set):

ACCURACY 100.0% LOSS 0.00

Confusion matrix (validation set):

	NOISE	XIAORUI
NOISE	100%	0%
XIAORUI	0%	100%
F1 SCORE	1.00	1.00

Metrics (validation set): Resume tutorial

Through multiple iterations, model generalization can be improved.

Verified improvement methods include: optimizing the test dataset, adjusting the model structure(e.g., adding more convolutional layers), etc.

The platform has training time limits; free users can only train for up to 20 minutes;

If unlimited training time is needed, a paid service must be purchased.

6 Model Evaluation

Steps:

1. Click on the left **Model testing**.
2. Then click **Classify all**.
3. Begin classifying all the test set data.

The screenshot shows the Edge Impulse Studio interface. On the left, the 'Model testing' option is selected in the sidebar. In the main area, the 'Test data' table is visible, and the 'Classify all' button is highlighted with a red box. The 'Model testing output' panel on the right displays the results, including a 100.00% accuracy and a confusion matrix.

SAMPLE NA...	EXPECTED OUT...	LENG...	ACCURACY	RESULT
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui

METRIC	VALUE
Area under ROC Curve	1.00
Weighted average Precision	1.00
Weighted average Recall	1.00
Weighted average F1 score	1.00

	NOISE	XIAORUI	UNCERTAIN
NOISE	100%	0%	0%
XIAORUI	0%	100%	0%

4. After testing, you can view the test results in the **Result** panel on the right.

The detailed view of the test results shows the 'Test data' table with 5 samples. The 'Results' panel displays 100.00% accuracy and a confusion matrix. The 'Feature explorer' panel shows a scatter plot of features.

SAMPLE NA...	EXPECTED OUT...	LENG...	ACCURACY	RESULT
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
xiaorui_01...	xiaorui	1s	100%	1 xiaorui
noise_019	noise	10s	100%	19 noise
noise_009	noise	10s	100%	19 noise
noise_011	noise	10s	100%	19 noise

METRIC	VALUE
Area under ROC Curve	1.00
Weighted average Precision	1.00
Weighted average Recall	1.00
Weighted average F1 score	1.00

	NOISE	XIAORUI	UNCERTAIN
NOISE	100%	0%	0%
XIAORUI	0%	100%	0%
F1 SCORE	1.00	1.00	0%

Feature explorer

noise - correct
xiaorui - correct

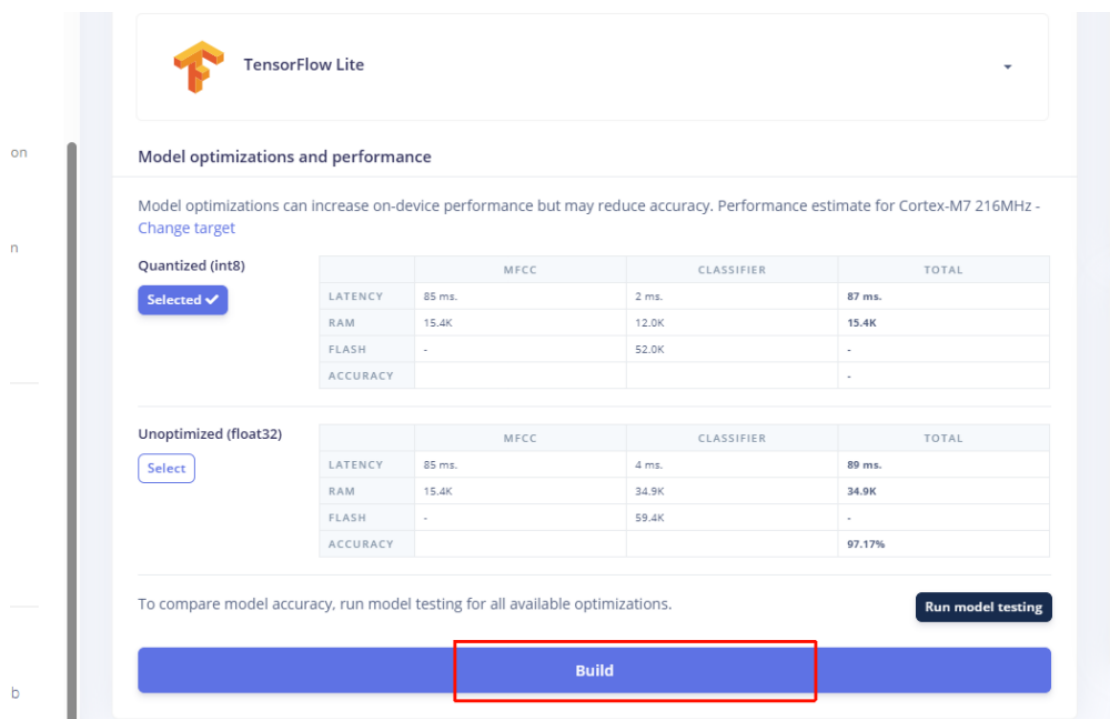
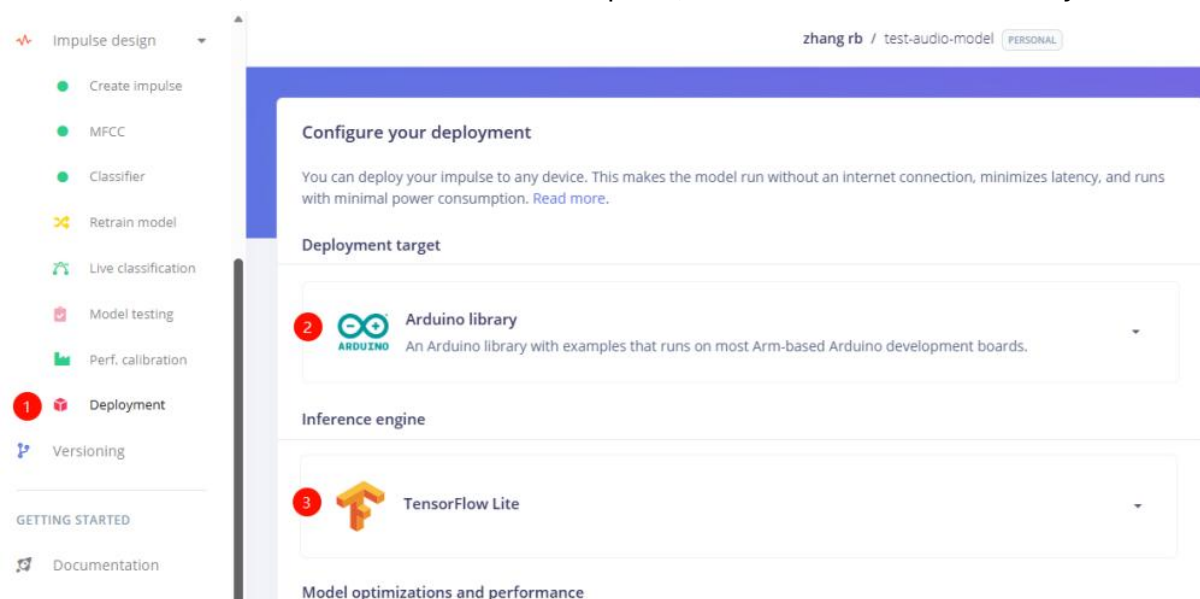
7 Model Integration

7.1 Generate Model Library File

After the model training is complete, we need to generate a library file that runs on the Edgi-Talk platform.

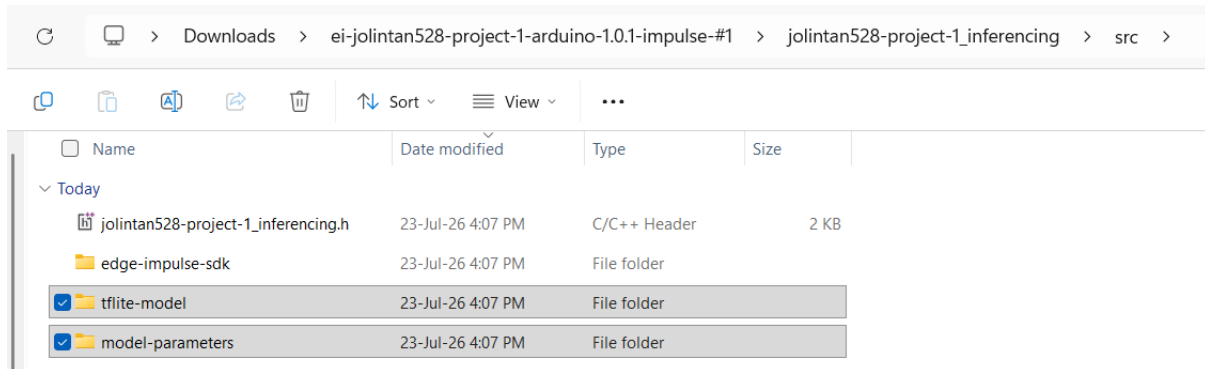
Steps:

1. Click on the left Deployment.
2. Search for Arduino library and TensorFlow Lite.
3. Then click Build, and once the build is complete, save the downloaded library file.

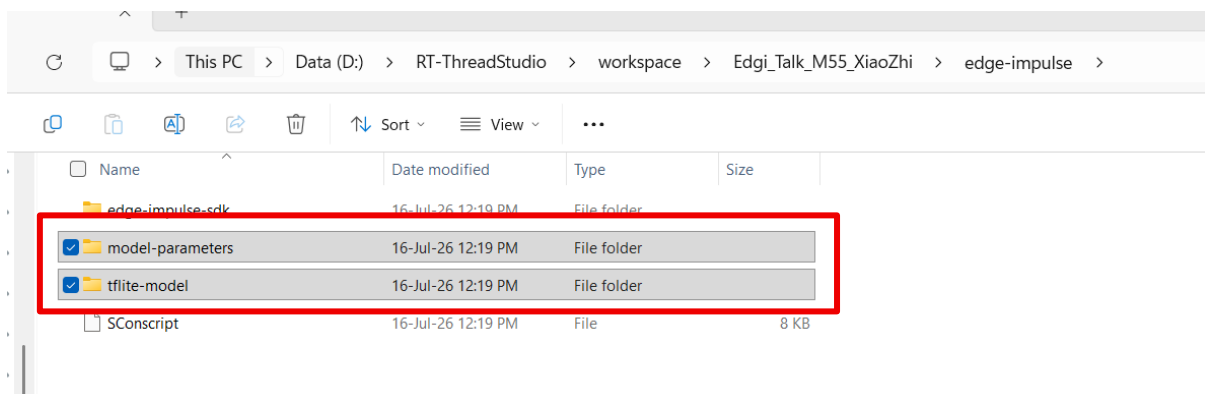


7.2 Deploy to Embedded Hardware

1. Extract the zip file downloaded from the previous step to get the following file directory.



2. Get the latest Edgi-Talk dev board SDK, go into the Edgi_Talk_M55_XiaoZhi/edge-impulse directory, delete the two folders shown in the image below, then replace them with the two folders from the previous step to replace the model files.



3. Compile the Edgi_Talk_M55_XiaoZhi project and flash it onto the dev board.

4. In the Edgi-Talk serial terminal, enter the following command to start the wake word test:

```
xz_wakeword_init
```

```
xz_wakeword_start
```

5. Say the wake word to the dev board; the demo here uses "xiaorui "; when the wake word is spoken, the serial terminal will print a log

```
msh />xz_wakeword_start
[D/xz.wakeword] Wake word already enabled
msh />[W/xz.wakeword] *** WAKE WORD DETECTED: xiaorui (99.22%) ***
[D/xz.ws] Wake word detected: xiaorui (confidence: 99.22%)
[D/xz.ws] Playing wake sound
```

Note: If cannot type anything into the terminal, try



1. Opened the M33 Blink LED project's settings

- Expanded Edgi_Talk_M33_Blink_LED in Project Explorer → opened **RT-Thread Settings**.

2. Enabled the CM55 core

- Went to **Hardware** tab → **SOC Multi Core Mode** → checked "**Enable CM55 Core**".
- This is the critical setting — without it, the M33 firmware boots but never hands off execution to the M55 core, so your already-flashed XiaoZhi app in external flash just sits there unused.

3. Built the M33 project

- Set it as active project → built it → got a fresh rtthread.hex in Edgi_Talk_M33_Blink_LED\Debug\.

4. Flashed it via ModusToolbox Programmer

- Loaded that new M33 hex file (same Programmer GUI settings as before: External Memory ticked, PSE84_SMIF.FLM flashloader).
- Clicked Connect → Program. This wrote to the **internal flash bank** (cat1d.cm33.main_ns, 0x22... region) — the part that was never touched when you'd only flashed M55 before.

5. Did NOT need to reflash XiaoZhi again

- Because M33 and M55 live in separate flash regions (internal vs. external QSPI), your earlier XiaoZhi flash was still intact. Flashing M33 with CM55 enabled just gave it the "green light" to boot into the M55 app that was already sitting there.

6. Reset the board

- Power-cycled / reset → this time the boot sequence ran correctly: Secure M33 → M33 → **M55 started**, XiaoZhi UI initialized, msh /> prompt appeared.